CS 1337 Computer Science I
Fall 2023 Syllabus UTD

INSTRUCTOR INFORMATION

Name	Jason Smith
E-mail Address	smith@utdallas.edu
Office	ECSS 3.232
Office Phone	972-883-4835

Office Hours: M/W 10:00 AM – 12:00 PM or by appointment

All office hours will be held on MS Teams. Please use the link provided in eLearning and log into your Teams account. Guests will not be admitted.

Questions: All questions must be in person (class time and office hours) or posted to Piazza (http://piazza.com/utdallas/Fall2023/cs1337). I will not answer questions by email.

COURSE INFORMATION

Course Number	CE/CS 1337.002
Credit Hours	3
Meeting Time	Tu/Th 2:30 – 3:45 PM
Room	ECSW 1.365

DO YOU NEED ASSISTANCE?

Problem Solving Procedure

1. Try to solve it yourself – use the internet to research the problem and try different solutions. If you can't solve it after a couple of hours move to the next level.

2. Consult with your classmates – Post your question in Piazza or the class Discord server. On Piazza, you will be able to post your questions (anonymously if you wish) about anything related to the class and get a response either from me or a classmate. Private posts (that only I can see) should be reserved for grade related questions or questions involving large snippets of code. Since there are multiple people that can answer questions, you should get a quick response allowing you to complete the tasks you are working on. If your question can't be answered there, move to the next level.

3. Visit the CSMC – If your classmates can't answer the question, check with a mentor at the CSMC (https://csmc.utdallas.edu). If the CSMC mentors can't answer your question, move to the next level.

4. Visit me during office hours – If you are truly stumped, I will give you the information you need to move forward.

Help Desk: For help with issues regarding your computer, UTD maintains a walk-in help desk. Visit their Web site for details: http://www.utdallas.edu/ir/helpdesk/

If you need help, please make the effort to reach out. We can't help you if we don't know that you need help.

Resources:
- C++ language tutorial http://www.cplusplus.com/files/tutorial.pdf
- C++ reference: http://www.cppreference.com
- C++ tutorial http://www.learncpp.com/

WHAT DO I NEED FOR CLASS?

- **Zybooks (Required):**
  - CS 1337 – Computer Science I
  - You will pay for Zybooks when you enter the first assignment *(C-Strings and String Library Preview)*
  - All projects and homework assignments will be submitted through Zybooks for grading and plagiarism checking
  - You should receive a discount if you used Zybooks in 1336

- **Textbook (Required)**:
  - Starting Out with C++, From Control Structures through Objects
    (10th edition – colorful peppers); Gaddis, Tony; Pearson Publishing
    ISBN 9780137450626

    This is the same book that you may have used in 1336. It contains a few things not found in Zybooks. If you do not have this book already, you should be able to find it for very cheap as a used book. You may even try your luck at finding a digital version of it in the recesses of the internet.

- **C++ Compiler and IDE**
  - Zybooks contains an IDE for all of your coding assignments
  - I encourage everyone to use Online GDB (https://www.onlinegdb.com) for in-class assignments

- **Note Taking Supplies** – I expect you to actively take notes. Anything discussed in class or available in the textbook is fair game on a test. Have something available to collect all the knowledge given to you (be that in physical or digital form).

- **A computer** – Obviously, you need a computer for this class. Any computer bought off the shelf is sufficient for this class. However, if you do not have a computer, the CS computer lab is open 24/7 and contains all the software you will need for this class.

  I do not permit computers in class (except on in-class assignment days)**.** Tablets that use a digital pen (no keyboards or mice) are allowed for students to take notes. Exceptions will be made for those students that have accommodations from the Office of Accessibility.

  This is not a popular rule among students, but it is done so to maintain fairness in the academic environment for all students. There is no need to copy my examples in class as I will make them available to you. I would rather you focus on how we are solving the problem and how we are utilizing the C++ tools to help us.

REQUIRED COURSE INFORMATION SECTION

Course Prerequisite: Prerequisite: CS 1336 with a grade of C or better or equivalent (placement test).

Description of Course Content: Review of control structures and data types with emphasis on structured data types. Applies the object-oriented programming paradigm, focusing on the definition and use of classes along with the fundamentals of object-oriented design. Includes basic analysis of algorithms, searching and sorting techniques, and an introduction to software engineering.

Student Learning Outcomes: After successful completion of this course, the student should have an:

- Ability to use single and multi-dimension arrays.
- Ability to implement linear and binary searches.
- Ability to implement simple sorting algorithms.
- Ability to implement structured data types.
- Ability to define and implement a class.
- Ability to use fundamentals of object-oriented design.

Attendance Policy: Grade deductions will be applied based on the number of classes missed for the semester as noted below:

- 1-5 absences – No deduction
- 6-10 absences – 1 letter grade drop (10 point deduction)
- 11 or more absences – automatic F in the class
- 0 absences – maybe extra credit?

TEACHING PHILOSOPHY

CS 1337 is skills-based class. Like any skill, learning requires practice and perseverance. You don't learn to play guitar by strumming a few chords you saw in a YouTube video. It takes time, energy, and dedication to the craft. It also requires a change in mindset compared to high school. Many see CS as fun and exciting because of a high school class that just taught the basics and made simple games. But CS is more about solving problems and implementing those solutions in a foreign language. The projects you will create in this class are inspired by actual problems from the professional world. You may feel that CS is no longer "fun" once exposed to these problems.

The goal of the class is not to learn C++, but to learn the concepts associated with object-oriented programming and use those concepts to solve problems. Anyone can learn a programming language on their own, but the language is simply a collection of tools. What you do with the tools is where the learning takes place. Just because you know how to use a hammer doesn't mean you can build a house.

I teach the class using a reverse classroom approach. This means that most of the learning will happen outside of the classroom. The classroom is where we will come together to help fill in the gaps and answer questions based on your experience outside of class. You will learn the concepts by completing activities and writing code. It is expected that you will share information (not solutions) with your peers which will solidify your understanding of the material.

My goal is to prepare you for what lies ahead so you will not only be ready for your remaining classes, but also ready to transition into the professional world. It's never too early to think about the things you need for the next phase of your life after graduation. In addition to learning foundational computer science concepts, students will improve their critical thinking, communication,

debugging, and time management skills. All of these are necessary to be successful not only academically, but also professionally as well as in everyday life.

WHAT I EXPECT OF EACH STUDENT

- **Ask for help at any time.** If you do not understand something or are having trouble implementing a concept, follow the problem solving procedure defined on page 1. The sooner you ask that question, the sooner you will get an answer. That answer will allow you to move forward. I want you to succeed; don't be afraid to ask questions.

- **Take responsibility for your education.** I am not the type of teacher that lectures with slides and expects students to memorize them. I teach by creating program examples in class and asking you to participate in completing the code as well as ask questions about things you don't understand. I will give challenging assignments to push you toward learning concepts in-depth as well as developing critical thinking and core programming skills.

  Part of being a professional is learning how to teach yourself. I am going to guide you through the topics of the semester, but a significant portion of what you take with you to the next class will be things that you have learned on your own.

- **Practice time management skills.** Good time management is necessary for this class. All assignments (homework and projects) are designed to be worked on over a period of days or weeks. **I expect everyone to devote an average of at least an hour a day to this class (7-10 hours a week).** Doing this will help you to divide tasks up into chunks and work a little at a time on an assignment rather than waiting until a day or two before it is due. You will have a very difficult time succeeding in this class if you schedule to finish every assignment at the last minute.

- **Attend every class.** You are paying for an education. Don't waste your money by skipping class. I will give you everything you need to complete projects and do well on the tests. You have to be there to get the information.

- **Make mistakes!** This is how you learn. Don't be discouraged when something goes wrong. Programming takes lots of practice and mistakes will always happen. Study the mistakes you made so that you can learn from them for the future.

WHAT EACH STUDENT SHOULD EXPECT

- **A problem solving class.** This class is not a programming class. Computer science is all about problem solving. The content of this class is to teach you how to solve problems using a computer. In order to solve those problems, you will need to learn a foreign language (C++) and write solutions that the computer can interpret.

- **An open environment dedicated to learning.** I want students to feel free to voice their opinions. Oftentimes as we code in class, I will ask students what they would do in a certain situation. I want each student to feel as if he/she can speak freely and also be open for other students to discuss that idea, even if that means that some students will disagree.

- **Exams focused on syntax and application.** The programming language and how to use it are connected; each one is as important as the other. In the same way that knowing how to spell words does not make you a good writer, knowing the syntax of C++ does not mean that you know how to use those tools correctly. I will test both of these things in the exams. You

will need to know the proper syntax for the programming concepts and how to use them properly. You will be expected to write code on the exam.

- **A simulated professional experience.** The projects in this class require you to exercise strategies found in "the real world". Your logic for a project may force you to learn new techniques that haven't yet been discussed in class. It is very likely that you will discuss logic with your classmates to better understand why your program isn't working. You will have to perform code maintenance and improve the efficiency of previously written code. These things offer a small taste of how life might be once you graduate and are given large sums of money by a company seeking your skills.

- **A deep understanding of C++ and object-oriented programming.** My goal is for you to know all the topics of CS 1337 so well that you do not need to look them up when trying to use them. I'm going to push you toward that goal. You should have peace of mind moving on in your degree plan because you will be fully prepared to tackle what the next course in the sequence will throw at you.

THE INFORMATION YOU REALLY CARE ABOUT

Grading Scale:

98-100 A+	88-89 B+	78-79 C+	68-69 D+	Below 60 F
92-97 A	82-87 B	72-77 C	62-67 D	
90-91 A-	80-81 B-	70-71 C-	60-61 D	

Grade Components:

Projects (4)	40%
Exams (3)	30%
Preview Homework	10%
Review Homework	10%
In-class Assignments	10%

General Grade Information: All grades will be available in eLearning. The Weighted Total column will give you the most accurate information concerning your grade. The weighted total is an approximation of your grade in the class based on the grades currently in eLearning.

I do not curve grades. Assignments are combined into categories so that a low grade for one item will not destroy your grade. There are things built into the course to help students obtain higher grades such as multiple submissions for assignments and exam corrections.

Grade Disputes: All grade disputes must be reported within 1 week and resolved within 2 weeks of the grade in question being posted in eLearning. Uncontested grades will become final after 1 week and cannot be disputed later. Announcements are made after each grade is posted so please check your grades promptly and reach out to the proper person.

I am responsible for grading your exams. If you have questions regarding your exam, please contact me through a private post on Piazza.

Everything else will be graded by a TA. Please address any grading concerns you have regarding these grades with the TA. **When you email the TA with questions about your grade, copy me on the email so that I am aware of the situation and can make sure it is resolved.**

You are responsible for verifying that all grades in Zybooks are recorded in eLearning correctly. Once you submit your assignment in ZyBooks, remember to press the Submit to

Blackboard button. Then, check My Grades in eLearning to check that the correct grade was transferred.

Projects: Projects will be major programming assignments that supplement recently discussed topics and will be completed in two to three weeks. Projects are intended to take approximately 15-20** hours to complete overall; this includes the design, coding and testing process. Waiting until a couple of days before the due date to start the project is a bad idea. Not only does this introduce unnecessary stress into your life, it hardly ever ends well for the student. Most students score poorly on projects that are built in less than three days.

*(** this is average time for a student that has taken 1336 at UTD. Your experience with problem solving in computer science may impact the time needed to complete assignments)*

Projects will be divided into milestones as you would expect to see in the professional world. The milestones for each project are as follows:

- Design (4 days)
- Core implementation (7-10 days)
- Final implementation and testing (7-10 days)

Each milestone will be graded and the total of all three milestones will contribute toward the overall grade for the project. Each milestone will have a firm deadline and failure to meet the deadlines will have a negative impact on your grade.

Projects are individual endeavors and students are not to work in groups on any project. Students are permitted (and I openly encourage students) to discuss the project. Feel free to share ideas on the logic, but **DO NOT SHOW YOUR CODE TO OTHER STUDENTS.** When discussing logic, try to keep it general. If you give out every little piece of logic you have, there is a good chance the person you are helping will have very similar code as yours and may be flagged for being too similar. Be careful of posting your code online. Another student could use your code without your knowledge and could involve you in a code plagiarism referral.

Students should avoid using web sites like GitHub and Chegg for help on projects. Copying code from a web site is considered plagiarism and will be treated as such. If you find code on a web site, it is highly likely another student will find it as well which may cause both submissions to be flagged for similarity. Submitting project and assignment details to web sites for outside help (e.g. Chegg) is also considered academic dishonesty by UTD.

All projects will be submitted in ZyBooks and will be compared for originality. Any projects that are approximate or identical copies will be reported to the Office of Community Standards and Conduct, and I will accept their decision in regard to the grade if they believe that academic dishonesty has occurred.

Programming assignments will be graded on a 100 point basis. Not only will your project be graded on proper execution, but also things like efficiency, implementation and documentation. Keep in mind that you always want to write code that is easy to understand and is also easy to maintain. Fewer lines do not necessarily mean a better program. Please use comments liberally.

You are responsible for testing your project thoroughly before submission. I will not give you the exact test cases that will be used for grading before the project is due. As a computer scientist, you must be able to identify all possible input and make sure that your code produces proper output and does not crash.

Late Projects:

All project milestones will be due at 11:59 PM on the day listed in the project documentation. The final project code will be accepted up to 12 hours late with the following penalties

<= 1 hour	-5 points
1 – 3 hours	-10 points
3 – 12 hours	-25 points
12 – 24 hours	-50 points

Preview Homework: Preview homework assignments are activities based on the readings in ZyBooks. Students will read through the sections and answer basic questions about the material to illustrate they understand the general concepts.

Review Homework: Review homework assignments are generally short coding assignments that can be done in 1-2 hours and measure how well you understand the material we have covered. These assignments are typically due 1 week from the date given.

Exams: Exams will cover chapters as listed below in the tentative course schedule. Exams will include a variety of question types including multiple choice, multiple answer and essay questions. Students are expected to be able to apply knowledge from all previous chapters in conjunction with the tested chapters. Exams are not created to make you feel smart; they are designed for you to demonstrate your understanding of the concepts. A high score on an exam exhibits a deep understanding of the topics.

An exam should not be missed except for the most extreme. If you miss an exam, you must have documentation for the absence. A make-up exam may be given to students with valid documentation. The allowance of a make-up exam is at the sole discretion of the instructor.

All exams will be given online in eLearning. Students are expected to take the exam at the Testing Center on the day scheduled during the window of opportunity provided by the testing Center. The exams will be closed book and closed notes. All solutions to coding problems on the test will be submitted for similarity to maintain academic integrity.

You are expected to be at the Testing Center at your scheduled time. You must start the exam no later than 10 minutes after your scheduled time. Failure to do so will result in a 10 point penalty on the exam.

Assignment Due Date Exceptions: In general, assignments are not accepted late except for the final submission of a project (see late penalties above). However, I know that life has a way of bringing the unexpected at the most inopportune times. If you have an exceptional life situation (personally or academically) that is creating difficulty for you to meet the given due date of an assignment, I will work with you to give an extension as long as you contact me **at least 24 hours before the due date**.

ARE WE THERE YET?

All dates are subject to change at the discretion of the instructor.

Date	Topic	
8/22	Introduction to C	C++ Concepts in C (eLearning)
8/24	Character and C-string Functions	Chapter 10 (Gaddis) Chapter 1 (Zybooks)
8/28	**Last day to add/swap classes**	
8/29	Characters, Strings, and the String Class	Chapter 10 (Gaddis)

		Chapter 1 (Zybooks)
8/31	Advanced File I/O	Chapter 12 (Gaddis) Chapter 3 (Zybooks)
9/5	Advanced File I/O	
9/6	**Last day to withdraw without "W"**	
9/7	Recursion	Chapter 19 (Gaddis) Chapter 4 (Zybooks)
9/12	Recursion	
9/14	**In-Class Assignment #1**	
9/15	**Exam 1 (Chapters 10, 12, 19)**	
9/19	Pointers	Chapter 9 (Gaddis) Chapter 6 (Zybooks)
9/21	Pointers	
9/26	Smart Pointers	Chapter 9 (Gaddis) C++11 Smart Pointers (eLearning)
9/28	**In-class Assignment #2**	
10/3	Structures	Chapter 11 (Gaddis) Chapter 8 (Zybooks)
10/5	Linked Lists	Chapter 8 (Zybooks) Linked list resources (eLearning) Chapter 17 (Gaddis)
10/10	Linked Lists	
10/12	**In-class Assignment #3**	
10/17	Enumerated Data Types Pointers in C	Chapter 11 (Gaddis) Chapter 8 & 9(Zybooks)
10/19	Sorting Arrays	Chapter 8 (Gaddis) Chapter 10 (Zybooks)
10/24	Searching Arrays	
10/26	Introduction to Classes	Chapter 13 (Gaddis) Chapter 12 (Zybooks)
10/27	**Exam 2 (Linked Lists, C. 8, 9, 11)**	
10/31	Introduction to Classes	Chapter 13 (Gaddis) Chapter 12 (Zybooks)
11/2	Copy Constructor Overloaded Operators	Chapter 14 (Gaddis) Chapter 12 (Zybooks)
11/7	**In-class Assignment #4** **Last day to drop**	
11/9	Overloaded Operators	Chapter 14 (Gaddis) Chapter 12 (Zybooks)
11/14	Overloaded Operators	
11/16	**In-class Assignment #5**	
11/20- 11/24	**FALL BREAK**	
11/28	Static members	Chapter 15 (Gaddis)

	Inheritance	Chapter 13 (Zybooks)
11/30	Inheritance Polymorphism	
12/5	Polymorphism Virtual Functions	
12/7	**In-class Assignment #6**	
12/12	**Exam 3 (C. 13, 14, 15)**	

Assignment Calendar

All assignments due by 11:59 on the due date listed unless otherwise noted

All dates are subject to change at the discretion of the instructor.

Please refer to eLearning for any updates.

All submissions are made using the provided links in eLearning

Assignment	Post Date	Due Date
C-Strings and String Library Preview	8/22	8/27
Review Homework 1	8/24	8/30
Advanced File Operations Preview	8/22	9/1
Project 1 Pseudocode	8/29	9/2
Recursion Preview	8/24	9/6
Review Homework 2	8/31	9/7
Project 1 Core Implementation	8/29	9/11
Review Homework 3	9/7	9/13
Exam 1	9/15	9/15
Project 1 Final Submission	8/29	9/17
Pointers Preview	8/22	9/19
Project 2 Pseudocode	9/21	9/25
Project 2 Core Implementation	9/21	10/1
Structures and Enumeration Preview	8/22	10/3
Project 2 Final Submission	9/21	10/8
Review Homework 4	10/3	10/11
Project 3 Pseudocode	10/10	10/14
Pointers in C Preview	8/22	10/16
Project 3 Core Implementation	10/10	10/21
Searching and Sorting Preview	8/22	10/23
Review Homework 5	10/19	10/25
Exam 2	10/27	10/27
Project 3 Final Submission	10/10	10/29

Objects and Classes Preview – Part I	8/22	10/31
Project 4 Pseudocode	10/31	11/4
Review Homework 6	11/2	11/8
Objects and Classes Preview – Part II	8/22	11/11
Review Homework 7	11/9	11/15
Inheritance and Polymorphism Preview	8/22	11/27
Project 4 Final Submission	10/31	12/1
Review Homework 8	11/30	12/6
Exam 3	12/12	12/12

University Policies
For all other University policies, please visit http://go.utdallas.edu/syllabus-policies